10/550090
JC05 Rec'd PCT/PTO 21 SEP 2005

4/PRTS

Description

Dialogue Learning System Based On Template-Template Structure

Technical Field

The present invention relates to a new dialogue learning system having a template-template structure based on extraction rules and exploiting the expanding power of buggy rules.

Background Art

The invention is motivated by a keen need for automating and simplifying the time-consuming authoring task used for language-oriented intelligent learning systems. This is because even though the number of possible correct right responses is reasonably limited, it is often necessary to deal with a large, in fact, theoretically unlimited number of plausible errors of learners for developing an ideal learning system. As far as the inventors can judge, at least within the foreseeable future, the state of the art in natural language processing technology cannot reach the level capable of providing a ready solution to an automatic correction of an entirely free format error-ridden sentence. This seems possible only when the world knowledge base of so-

called common sense may be introduced into the system so that many competent human teachers can somehow manage to cope with it.

Disclosure of the Invention

A learning system according to the invention ("Azalea"), to which the concept of template automaton is introduced, collects many "expected examples of a variety of learners, including "right" and "wrong" responses. As an efficient error diagnosis engine in the language learning system, the NLP technology of an HCS (heaviest common sequence) or an LCS (longest common sequence) algorithm plays a decisive role. Those examples embedded into the template is used for the diagnosis and analysis of learners' responses. The diagnosis is to be implemented by selecting a closest path from among a huge number of candidate paths in template databases to the learner's input sentences. The authoring task of building a template corpus consisting of well-formed model translations and ill-formed erroneous sentences is quite labor-intensive, taking up considerable time.

The new system of the invention not only simplifies or reduces the authoring task of template generation, which otherwise is time-consuming (refer to, JP-A No. 2002-49617 by Naoyuki Tokuda, Liang Chen, Hiroyuki Sasai, et al), but also

is effective for the improvement of system performance. The first reason why the introduced template-template architecture can simplify the system and improve the performance is that the architecture makes it possible to integrate many different templates into a single template-template and vice versa, that is to extract many different templates therefrom by applying extracting rules assigned to some of transition nodes of a single template. The second reason is that the introduced buggy rules have the function of automatically distinguishing and classifying erroneous learners' responses and, accordingly, generating bugs therefrom. The importance of the NLP (natural language processing) techniques in the development of the system is obvious because a parser is used for examining the learner's structure free-format response and the semantic structure is examined by checking the learner's response in character string against the semantically equivalent path of the provided template data base.

The template-template structure based on the new extracting rules or buggy rules is expected to play an important role in many applications, when used in any system selected from learning systems having character input and interaction means, voice-based call centers or voice-enabling portal systems, and any systems focusing on more enhanced

human computer interfaces implementing more natural human-computer interactions between the system and humans.

The present invention provides the following three contributions:

1. Embedding extracting rules into certain nodes of the template-template transitional diagram with certain constraints imposed in selecting the nodes enables a single template-template to represent a variety of different types of existing templates.

2. Exploiting the expanding power of the buggy rules introduced, many erroneous expressions and/or many erroneous syntactic structures can now be described automatically with consistency so that the new template-template can be expanded to a "thick" template-template form, because of its automatic capability of describing the bug taxonomy. The language teacher need not be concerned with details of classifying erroneous translations at the time of authoring the template-template, thus bearing a reduced burden of error taxonomy, when evaluating and clustering the characteristics of the particular errors of learners.

3. The HCS matching algorithm can be developed so that the algorithm matches the input sentence against the simpler template-template directly, thus reducing the spatial and temporal calculation load in matching processes in finding

the best matched paths from among all the possible path of all the extracted templates without actually expanding the template-template.

Best Mode for Practicing the Invention

Template-Template Structure

First the term "template-template" will be defined below. The template-template is defined as a special template where some of the nodes are marked with extracting rule-associated symbols allowing the template-template to be expanded into many templates, or a so-called large template if a set of non-connected templates are regarded as one template. Such a set of disconnected templates can now allow a variety of possible translations of a single L1 sentence to form a large single template-template comprising a group of translated L2 sentences. Being an extended template, the template-template scheme allows the template-template to extract one or more of the templates.

Typically, an extracting rule is always associated with a set of symbols, say $\{s_1, s_2, \ldots, s_n\}$, and each of the symbols is assigned one or more nodes in the templates. These associated symbols are assigned with one or more values whose function is to represent the style of the nodes that will appear in a template or templates extracted from the

template-template. These symbols are herein referred to as "label symbols." The symbols related to a single rule are called "related symbols." Related symbols should have certain restrictions. As a typical restriction, for a given $s_i=1$, s_k must often be restricted to 2, or to some positive integers other than *1*. If the value of s_i depends on the values assigned to a set of the other symbols, the choice of value of s_i is called a required choice of the other symbols.

Some examples of the extracting rules are given below so that language teachers can easily understand.

Type A Rule AP (appear) - NAP (not appear) Rule

Suppose that some nodes are marked with AP*i*, while some other nodes are marked with *NAPi* (*i* being any integer, representing different Type A Rules). The AP-NAP Rule of Type A Rule imposes the condition that when expanded, a new expanded template can include either the nodes marked with AP*i* or the nodes marked with NAP_i, but not both of them at the same time. "$AP_i=0$" is used to denote the case where the nodes marked with AP_i do not appear in a template. At this time, NAP_i have to be *1*, meaning that the nodes marked with NAP_i will appear in the template. Thus, it can be seen that $NAP_i=1$ is the required choice of $AP_i=0$. By the same reasoning, when $NAP_i=0$, AP_i must have a value of *1*, so that $NAP_i=0$ is the

required choice of $AP_1=1$.

Type B Rule PPR (personal pronoun) - PPRP (personal pronoun possessive) Rule

As in the Type A Rule, the template-template rule imposes the condition that the nodes marked with *PPRi* and the other nodes marked with *PPRPi* (*i* being any integer) appearing in a set of templates must respectively take on the form of the personal pronoun and the personal pronoun possessive form of the pronouns as required by the natural language grammar of the pronouns. Given $PPRP_1$ (or PPR_1), the required values of PPR_1 (or $PPRP_1$) must be defined by the natural language grammar of pronouns.

Type C Rule AN (arbitrary number) Rule.

Type C Rule imposes the condition that any positive real number can be assigned to the nodes marked by AN_1 . If it is true that "I have 5 books on Zen," this Rule AN_1 can be assigned to the error node of 5, because any number other than 5 is erroneous.

Type D Rule CHO (choosing-one) rules.

A type D Rule imposes the condition that among all the nodes of the template-template marked by $CHOi_1$, $CHOi_2$, ..., $CHOi_k$,, one and only one set of nodes can appear in any of the

templates extracted from the template-template. Here a different i represents a different Type D Rule. Hence, $CHOi_j=0$ implies that the nodes marked by $CHOi_j$ do not appear, while $CHOi_j=1$ implies that the specified nodes now appear. Obviously, if 1 is assigned to one $CHOi_k$, then 0 should be assigned to all the other $CHOi_j$.

The Buggy Rules for Expanding Template-Template

A buggy rule here is defined as a production rule of common syntactically erroneous expressions which are characterized by possible deviations from syntactically correct expressions.

To be specific, consider the following form of a buggy rule:

$$H_1H_2...H_N \rightarrow R_1R_2...R_M$$

where $H_1H_2...H_N$ represents a set of nodes tracking the syntactically correct path of any template-templates, or a set of grammatical part-of-speech tags representing basic components or segments of a correct expression. $R_1R_2...R_M$ is the set of nodes which represents a typical erroneous expression whose correct form is $H_1H_2...H_N$. It is immediately seen that errors are identified by deviations from the correct paths of the template-template. Here is an example:

EX VBP→ EX VBZ

(Here EX represents existential such as "there is," VBP,

verb for 1st and 2nd person present, VBZ, verb, 3rd person singular present). This example implies that a syntactically correct expression "*there are* 5 books" is used erroneously by students who misunderstood the subject-verb alignment, resulting in an erroneous expression of "*there is* 5 books" in this example.

Brief Description of the Drawings

Figure 1 is a diagram showing a template-template structure.

Figure 2 is a diagram showing a template-template structure expanded by expansion rules.

Figure 3 is a diagram showing a template 1 expanded for a sentence meaning "Japan is dotted with beautiful gardens nationwide".

Figure 4 is a diagram showing a template expanded for a sentence meaning "Japan is dotted with beautiful gardens nationwide".

Description of Symbols

Error Messages:

AS: an assumption has been made on the quantity of the noun

AT: the article is not needed

CM: a comma is needed

CT: contraction is incorrect

MN: meaning is incorrect

NP: noun must be plural

VS: verb must be singular, since subject is singular

PR: preposition is incorrect

PP: phrase must be plural

Typical Part-of-Speech Tags:

DT: Determiner	EX: Existential
IN: Preposition/Subord. conjunction	JJ: Adjective
NN: Noun, singular or mass	NNS: Noun, plural
NNP: Proper noun, singular	RB: Adverb
VBN: Verb, past particle	VBP: Verb, non-3rd ps. sing. Present
VBZ: Verb, 3rd ps. sing. Present	

The invention will be described in the following embodiment with reference to drawings.

Figure 1 is a diagram showing a template-template structure; Figure 2 is a diagram showing a template-template structure expanded by the expansion rules; Figure 3 is a diagram showing a template 1; and Figure 4 is a diagram showing a template 2.

Example of Template-Template, Template-Template Expanded by Buggy Rules, and Templates Extracted from Template-Template

In the embodiment of the invention, a template-

template for English translations of a Japanese sentence meaning "Japan is dotted with beautiful gardens nationwide." is constructed at first. The numeral shown in Figure 1 and other figures represent the weights of word for enhancing the relative importance of the respective words in the sentence. The default weights of the words in the template are set to *1*, and they must be assigned in accordance with the importance of the words as judged by experts in the field. Reference is made to JP-ANo.2002-49617 by Naoyuki Tokuda, Liang Chen and Hiroyuki Sasai, for a detailed explanation. The symbols within "[" and "]" are the part-of-speech tags. The nodes shown in the left edge are starting nodes.

Now simply applying the buggy rules listed above can expand it into the template-template of Figure 2.

This shows that a language teacher need not be concerned with the details of classifying many common errors when he/she is constructing the template-template, since the buggy rules can generate taxonomization of bugs, automatically allowing these erroneous expressions to be built into the template-template.

Now, by applying the Type A rule, it is easy to see that it is possible to extract a template as shown in Figure

3 from the template-template of Figure 2 by allowing the nodes marked with AP_1 to appear in the template and accordingly by deleting the nodes marked by NAP_1 of Figure 2, as well as the template as shown in Figure 4, this time by deleting the nodes marked by AP_1 of Figure 2, and accordingly letting the nodes marked with NAP_1 appear in the template.

It can be seen that a language teacher is able to construct the template-template, integrating a large combination of templates in terms of simpler label symbols.

Matching Algorithm for Template-Template and the heaviest common sequence for an input sentence

As is evident from the above discussion, many templates can be extracted from a single template-template. Suppose a template-template has label symbols $s_1, s_2, \ldots, s_n$ to be associated with certain nodes of the transitional diagram; it is seen that the different templates extracted from the template-template can be obtained by assigning these symbols to nodes. In the invention, it is possible to denote each template extracted from the template by an *n-tuple* $\{s_1, p_1, s_2, p_2 \ldots, s_n, p_n\}$, where p_i's are proper assignment to symbols s_i. As has been discussed in an earlier section, p_i's can be either numbers or words in accordance with the extracting rules used.

A heaviest common sequence of two sentences is defined as an ordered sequence of words, $a_1, a_2, \ldots, a_m$ which appear in both of the sentences in an order of a_1 *then* a_2 *then* … *then* a_m . The definition of common sequence can be found in the book "*Foundations of Computer Science*" by A. V. Aho and J. D. Ullman (Computer Science Press, 1992, pp. 321-327)

As each word or phrase in the template is assigned with weights, the heaviest common sequence of a path in the template and an input sentence is defined as the common sequence among all the possible common sequences whose total weights are the largest.

A heaviest common sequence in words and/or phrases of an input sentence is searched for from among all the possible valid paths of the template.

The heaviest common sequence of a template and an input sentence is defined as the heaviest common sequence of words which has the heaviest total weight among the heaviest common sequences, each of which is obtained from one path of the template and the input sentence.

Once the template-template has been obtained in an application involving a language translation learning system,

the next step is to match an input sentence to each of all the possible templates, and then choose the closest path. A detailed description on the DP(dynamic programming)-based matching procedure of a template to a sentence can be found in JP-A No.2002-49617 by Naoyuki Tokuda, Liang Chen and Hiroyuki Sasai.

In the method of the present invention, a closest path is found from among all the valid paths of the templates that could be extracted from template-template with extracting rules (but without buggy rules) directly, without physically extracting all the templates from the template-template. It is necessary to expand the buggy rules-embedded template-template first so that the template-template does not include any buggy rules before such a matching takes place. This can be performed in the steps of Figure 2 described above.

The first step needed in the algorithm is to convert a template-template into its dual figure of an acyclic weighted finite digraph (directed graph), by simply expressing each node of the template as one or several arcs in the graph, by adding arcs labeled with *0* weight for each empty node where applicable. Since the digraph is converted from the template-template, it contains many arcs associated with label symbols whose functions depend critically on the values assigned to

the symbols. Accordingly given one such digraph, a completely different template can be extracted if a different set of label symbols are assigned to arcs. That is to say, given such a digraph, it is possible to obtain many digraphs, each of which corresponds to a template that can be extracted from the template-template. The digraph extracted from template-template is hereinafter called a template-digraph.

The inventors now define a procedure of finding the heaviest common sequences from among the common sequences of the paths of all the digraphs and an input sentence as below.

The heaviest common sequence of the paths ended by any special node *N* in a digraph and an input sentence is defined as the sequence of words which has the heaviest total weight among all the heaviest common sequences, each of which is obtained from one path ended by *N* of the digraph and the input sentence.

Furthermore, let $N_1\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$ represent the paths of all the digraphs extracted from a template-digraph but ending at node N_1, where the symbol s_i is assigned with value p_i $(i=1,2,\ldots,n)$. An n-tuple $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$ is referred to as a label of node N_1. Here it should be assumed that there is no contradiction of the rules when s_i is set as

p_1, s_2 as $p_2, \ldots$, s_n as p_n. Such a label $\{s_1, p_1, s_2, p_2, \ldots, s_n, p_n\}$ is reffered to as a contradiction-free label.

The heaviest common sequence of a node labeled $N_1\{s_1, p_1, s_2, p_2, \ldots, s_n, p_n\}$ and an input sentence is defined as the heaviest common sequence of words which has the heaviest total weight among all the heaviest common sequences, each of which is obtained as the heaviest common sequence of one digraph extracted from the digraph-template with the nodes marked with the contradiction-free label $\{s_1, p_1, s_2, p_2, \ldots, s_n, p_n\}$.

Note that some nodes can not appear simultaneously in one digraph extracted from a digraph-template, as in a node labeled AP2 and a node labeled NAP2 in one digraph. As the result, the rule breaking label such as $N_1(\ldots$, AP2,1, …, NAP2,1…) should not be allowed in any calculation scheme of the common sequences of a node in the digraph-template and an input sentence. The following algorithm describes the procedure for calculating the heaviest common sequence of a template-template and an input sentence. In the following calculation, "λ" is used as a very special value of the label symbols, whereby its value remains unspecified up to a certain stage of the calculation.

1. Turn the template-template into a template-digraph of

which the directed edges (transitions) are labeled by the corresponding words in the template;

2. Topologically sort all the nodes of the digraphs into N_1, N_2, ..., N_t, such that for each pair of nodes N_i and N_j, there is no transition from N_j to N_i, when $j>i$;

3. Add an empty node N_0 to the digraph, and add an arc from N_0 to all the starting nodes in the template-template.

4. Set $CM(N_0,M_0)=0$

5. For $i=0$ to t, do the following steps:

6. If there is at least one arc to N_i which is associated with a symbol, and the s related labels would never exist after the node N_i, then for all i's, for j=0 to m, do the following:

Check all $CM(N_i\{\ldots\},M_j)$, for any label $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$, if no s-related label appears in $\{s_1,s,_2, \ldots,s_n\}$ and there exist at least an $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n, sx_1,px_1, sx_2,px_2, \ldots, sx_h,px_h\}$, where sx_1, sx_2, ...,sx_h are s related labels, such that $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n, sx_1,px_1, sx_2,px_2, \ldots, sx_h,px_h\}, M_j)$ is defined. Define $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}, M_j)$ as the largest

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,\ sx_1,px_1,\ sx_2,px_2,\ \ldots,\ sx_h,px_h\},\ M_j)$, and un-define all the $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,\ sx_1,px_1,\ sx_2,px_2,\ \ldots,\ sx_h,px_h\},\ M_j)$ that had already been defined.

7. For j=0 to m, do the following steps:

8. For each of the nodes N_k to which there is an arc from N_i, do the following:

(1) If arc N_iN_K has no label, Check all the $CM(N_i\{\ldots\},\ M_j)$, $CM(N_i(\ldots))))),\ M_{j+1})$, $CM(N_k\{\ldots\},\ M_j)$, $CM(N_k(\ldots),\ M_{j+1})$ that have been defined, define $CM(N_k\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_{j+1})$ as the maximum of the following data if one of $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$, $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_{j+1})$, $CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$, $CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_{j+1})$ has already been defined:

$CM(N_i\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$ if it has already been defined

$M(N_i\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)+W(N_iN_k)$ if $CM(N_i\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$ had already been defined and the arc N_iN_k is matched with W_k

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_{j+1})$ if it has already been defined

$CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$, if it has already been defined

$CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_{j+1})$ if it has already been defined

(2) If arc N_iN_k is associated with symbol s, check all the $CM(N_i(\ldots),\ M_j)$, $CM(N_i\{\ldots\},\ M_{j+1})$, $CM(N_k(\ldots),\ M_j)$, $CM(N_k\{\ldots\},\ M_{j+1})$ that have been defined.

(i) If the node label $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,\ \lambda\}$ is a contradiction-free label, and at least one of the following has already been defined:

$CM(N_1\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$, $CM(N_1\{\ s_1,p_1,s_2,p_2,\ldots,s_n,p_n\ \},\ M_{j+1})$,

$CM(N_k\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$, $CM(N_k\{\ s_1,p_1,s_2,p_2,\ldots,s_n,p_n\ \},\ M_{j+1})$,

$CM(N_1\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,\ \lambda\},\ M_j)$, $CM(N_1\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,\ \lambda\},\ M_{j+1})$,

$CM(N_k\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,\ \lambda\},\ M_j)$, $CM(N_k\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,,s,\ \lambda\},\ M_{j+1})$;

Define $CM(N_k\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,\ \lambda\},\ M_{j+1})$ as the maximum one of the above defined data.

(ii) Given label $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$, suppose that the label $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,p\}$ is a contradiction-free label, and that at least one of the following is true:

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$ has been defined and, if setting s to p is a required choice of $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$ or N_1N_k is matched to M_{j+1} after p is assigned to s.

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_{j+1})$, has been defined and if setting s to p is a required choice of $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$

$CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$, has been defined and if setting s to p is a required choice of $\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\}$

$CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,p\},\ M_{j+1})$, has been defined

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,p\},\ M_j)$, has been defined

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,,s,p\},\ M_{j+1})$, has been defined

$CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,,s,p\},\ M_j)$, has been defined

$CM(Nk\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,,s,p\},\ M_{j+1})$ *has been defined*

Define $CM(N_k\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,p\},\ M_{j+1})$ as the maximum one from the above defined data and the following data

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)+W(M_{j+1})$ if $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n\},\ M_j)$ is defined and N_iN_k is matched to M_{j+1} after p is set to s.

$CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,p\},\ M_j)+W(M_{j+1})$ if $CM(Ni\{s_1,p_1,s_2,p_2,\ldots,s_n,p_n,s,p\},\ M_j)$ is defined and N_iN_k is matched to M_{j+1} after p *is assigned to* s

And also change those s_i from λ to a required choice of the value, subject to the condition of p being assigned as s.

Among all the $CM(N_x,\ M_m)$ already defined with N_x being a final vertex, the largest $CM(N_x,\ M_m)$ will be the weight of the heaviest common sequence of the template-template and the path.

Note that, in the above algorithm, whenever *CM(N.(..), M.)* is selected from several candidates a kind of back link is set to the selected one . Note that by tracing the back link it is possible to obtain the path of the extracted template which has the heaviest common sequence with the input sentence immediately, as having found the weight of the largest common sequence of the template-template and the path.

Industrial Applicability

Although the invention has been described for the technical field of a natural language learning system, the application of the invention is not limited to natural language learning systems, and rather the invention is applicable to any of voice-enabling technology, a programming language learning system; and systems which need more natural advanced interfaces allowing human-computer interactions.